

# Plugins

Plugins are the only one way to change the functionality of a modeling tool. The main purpose of the plugin architecture is to add a new functionality to your modeling tool. Although there is a limited ability to remove an existing functionality using plugins.

A plugin must contain the following resources:

- A directory
- Compiled java files, packaged into a jar file
- A plugin descriptor file
- Optional files used by the plugin

Typically a plugin creates some GUI components allowing a user to use the plugin functionality. Generally, this is not necessary because the plugin can listen for some changes in a project and activate itself on the desired changes.

## Related pages

- [How plugins work](#)
- [Writing plugins](#)
- [Compilation classpath](#)
- [Testing plugin](#)
- [Detail information](#)
  - [Plugin descriptor](#)
  - [Plugin classes](#)
  - [Plugin class loading](#)
  - [Plugins directories](#)
- [Resource dependent plugin](#)
- [Creating test cases](#)
  - [Creating JUnit test case](#)
  - [Comparing projects for testing purposes](#)
  - [Working with test resources](#)
  - [Configure test environment](#)
- [Plugin integration with Eclipse](#)
- [Developing plugins using IDE](#)
  - [Development in Eclipse](#)
  - [Development in IntelliJ IDEA](#)